DAVID BRUBAKER,
CONTRIBUTING EDITOR

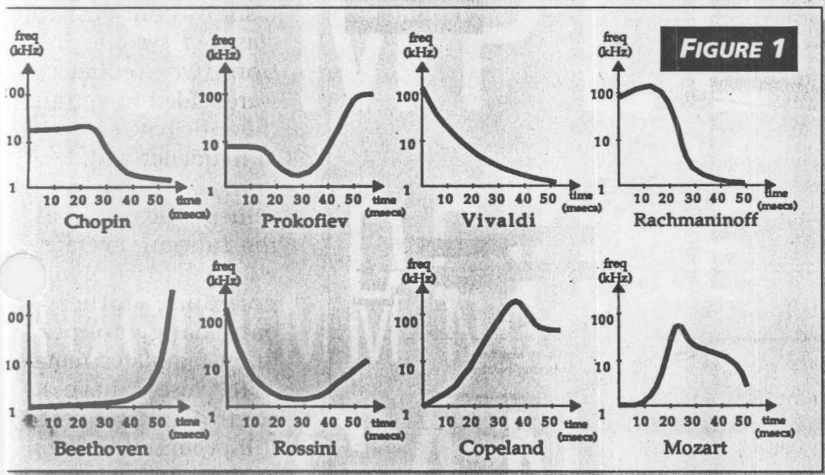
Fuzzy templates—with a twist

Every so often, I get caught off guard. Actually, it happens all the time in general living, but it also happens occasionally in engineering. Here's an example. I present it because it shows how fuzziness can be incorporated in a system without rules, but be prepared for a twist at the end that, I admit, I did not expect.

About two years ago, I helped develop a system to classify acoustic time-series "signatures." **Figure 1** shows plots of the eight possible signatures—acoustic "whistles" or "chirps," with the names of classical-music composers as code names. The vertical axis of each plot is frequency, which ranges from 1 to 100 kHz (note the log scale). The horizontal axis is time, in milliseconds. There are distinct and detectable start and stop times for each chirp, and no chirp lasts longer than 75 msec.

A different type of emitter generated each signature type. Upon receiving an unknown chirp, the system was required to classify the chirp into one of the eight emitter types. Not belonging to any of the eight designated types was also an acceptable outcome. Filters eliminated most noise prior to the classifier. However, based on emitter power and configuration, and specifics of the transmission medium, the classifier needed to accurately identify signatures with as much as 25% scaling in both frequency and chirp duration. Finally, future versions of the classification method would need to identify as many as 30 signal types.

FIGURE 1



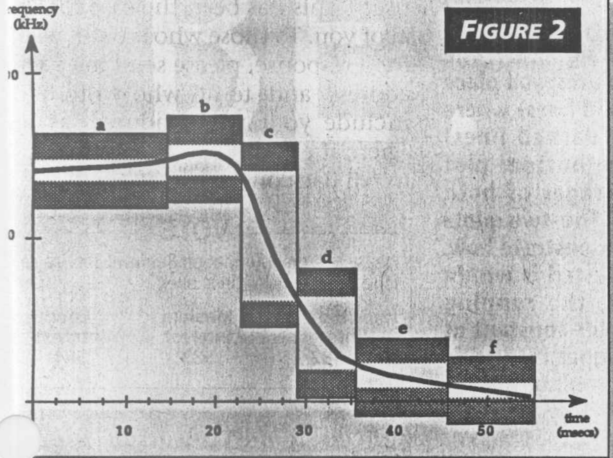
known incoming signatures must be matched to one of eight acoustic signature types, labeled with composers' names. The unknown signature must be ssified as being one of these eight, but the signature also might not match y of the types.

Initially ignoring the need to handle scaled signatures and because of the relative simplicity and small number of signature types, a simple template-matching scheme seemed appropriate. Traditionally, the principal problem with templates is that they are "brittle"; the transition from a perfect match to a complete mismatch tends to be rapid. I hoped adding fuzziness to the templates would greatly reduce or eliminate this brittleness.

Figure 2 shows a template for the signal type Chopin. It consists of six semifuzzy regions in the time-frequency input space. Each region is semifuzzy because it has fuzzy boundaries only for frequency; time boundaries are crisp.

Signature characteristics determine the placement of time and the frequency boundaries of each region. Selecting the number of regions in the template for each signal type involves a trade-off between the processing time and the ability of the template to accurately discriminate between different, but similar-looking, signature types. In general, you need more regions where the signature changes rapidly (where the magnitude of its slope is great), and you need fewer regions where it does not change rapidly (where the magnitude of its slope is small). **Figure 3** shows the membership functions, as functions of (log) frequency, that define regions *a* and *c* in the Chopin template.

FIGURE 2



semifuzzy regions in the frequency-time input space ake up the template for a Chopin-type signature. gions are semifuzzy because they are fuzzy for frequen- but they have crisp boundaries in time.

During operation, an incoming signature is sampled every millisecond. The sample value is applied to all active templates—to the fuzzy region within each template that corresponds to the time of the specific example. For example, when applied to Chopin's template, all samples occurring at time

0<t≤14 msec are applied to membership function *a*, all samples occurring at time 14 msec<t≤22 msec are applied to membership function *b*, and so forth.

How well each sampled frequency value fits into the respective template region determines the degree of match, μ. When the frequency sample is a full member of the template region, $\mu=1$. When it is *not* a full member of the region, $\mu=0$. Between these two values, $0<\mu<1$, and μ is an indication of how removed the actual value is from the required value.

Combining individual degrees of matching generates a measure of how well an actual signal matches a template. As a combining operator, the fuzzy AND operator is attractive because it may ease implementation but is inappropriate because it drives the result irretrievably to zero for any single time a zero match occurs. A running average is more appropriate. For each μ_i, the running average, $\bar{\mu}_1$, for all μ through μ_i is:

$$\bar{\mu}_i = \frac{(i-1)\bar{\mu}_{i-1} + \mu_i}{i}$$

Now for the twist. I wanted to see how much using fuzzy boundaries on the regions improved classifier operation, so I created a crisp-boundaried template for Chopin-type signatures and applied a signal that had a 20% longer-than-normal duration to both fuzzy and crisp templates. Figure 4 shows the results.

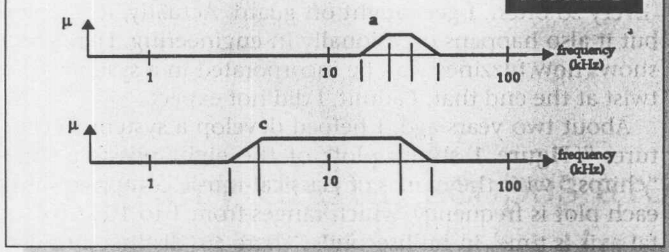
Fuzzy and crisp templates performed nearly identically. This result came as a complete surprise to me: I had expected the "softness" associated with the fuzzy boundaries to allow its overall degree of match, $\bar{\mu}_i$, to degrade far less rapidly. I ran the same comparison with other signal types and other templates, always with very nearly the same results.

A little thought provided the reason. Although we were using fuzziness, we were not using fuzzy logic, because we did not use a logic-based operator to combine the degrees of match. The running average, by providing smoothing on the bilevel logic measures, added the same improvement usually gained by moving to fuzzy logic. For this application, fuzziness was not needed for a successful system.

Did we, therefore, step back to a crisp implementation? No, we didn't. The actual matching mechanism was only part of the classifier system. In addition, two feedback loops were added to account for the anticipated 25% scaling in frequency and duration. Both loops had individual values of μ as inputs, not the running average, and we opted to work with proportional, rather than bang-bang, loops. The fuzzy templates remained.

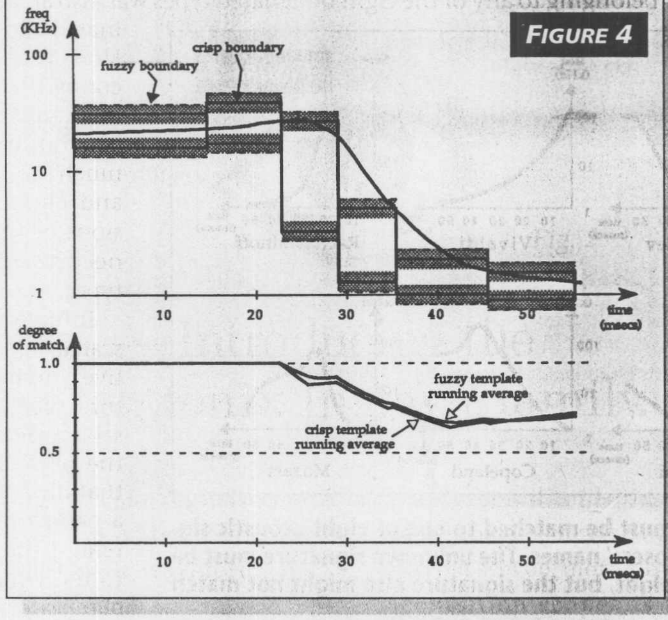
To close, I have a brief administrative matter. I still promise to respond to each reader who writes to me. Most of you write via e-mail, and I respond accordingly. Occasionally, a reader's server bounces back my response as addressed to an "unknown user." This has been the case for several of you. To those who are still waiting for a response, please send me a viable address, and, to all who write, please include your postal address as well. Then, if e-mail fails, I have something to fall back on.

FIGURE 3



These plots show two of the six regions of the Chopin-type template shown in Figure 2.

FIGURE 4



A template made up of regions with crisp boundaries performed as well as the template with fuzzy boundaries when processing a scaled signature. The top plot shows the templates; you place the crisp boundaries (solid lines) where the fuzzy boundaries (dashed lines) have values of 0.5. The bottom plot shows the running averages of both crisp and fuzzy values. The two plots correlate in time and demonstrate how, when the signal being tested is within the region boundaries, the running average increases (or holds constant at unity), and when the signal is outside region boundaries, the running average decreases.

VOTE

Please use the Information Retrieval Service card to rate this article (circle one):

High Interest	Medium Interest	Low Interest
582	583	584

David Brubaker is a consultant in fuzzy-system design. You can reach him at Huntington Advanced Technology, 883 Santa Cruz Ave, Suite 31, Menlo Park, CA 94025-4608 or on the Internet at: brubaker@cup.portal.com.